Project Plan Presentation
Smart Auto-Time Logging
The Capstone Experience

Team Vectorform

Everett Case
Sherwin Soroushian
Jake Mitchell
Jianyu Deng

Department of Computer Science and Engineering
Michigan State University
Fall 2021
Functional Specifications

• Optimize project billing process for Vectorform employees
• Improve efficiency for Vectorform employees by automatically tracking time worked on billable projects
• System will learn over time which project is being worked on
• User can interact with program through dynamic user portal
Design Specifications

• On start, program will begin tracking employee’s active work time on billable projects

• User can access user portal to show all time tracked for their projects with the ability to insert, delete, and update individual entries

• User will initially enter billing project code for unregistered entries, system will learn to assign new similar time entries to that project code

• User can confirm accurate project time through user portal and upload to Vectorform billing server
Screen Mockup: User Portal

![User Portal Mockup](image_url)
Screen Mockup: Date Selector

Figure 1

Figure 2
Screen Mockup: Editing

Figure 3

Figure 4

Figure 5
# Screen Mockup: Deleting

<table>
<thead>
<tr>
<th>Project Code</th>
<th>Start Time</th>
<th>End Time</th>
<th>Tracking Module</th>
<th>Meta Data</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>2556</td>
<td>8:25 am</td>
<td>9:45 am</td>
<td>Microsoft Teams</td>
<td>Meeting with execution team</td>
<td></td>
</tr>
<tr>
<td>112233</td>
<td>10:00 am</td>
<td>11:00 am</td>
<td>Microsoft Teams</td>
<td>Meeting with applications team</td>
<td></td>
</tr>
<tr>
<td>112233</td>
<td>11:15 am</td>
<td>2:00 pm</td>
<td>Visual Studio Code</td>
<td>ReactJS</td>
<td></td>
</tr>
<tr>
<td>112233</td>
<td>10:25 am</td>
<td>1:30 pm</td>
<td>Google Chrome</td>
<td></td>
<td></td>
</tr>
<tr>
<td>112233</td>
<td>3:30 pm</td>
<td>5:00 pm</td>
<td>Microsoft Teams</td>
<td>Happy Hour and Trivia</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6**

**Figure 7**
Screen Mockup: New Entry

Figure 8

Figure 9
Technical Specifications

• On start Python will begin tracking projects
  ▪ Webhook for Microsoft Teams API for communications
  ▪ Polling operating system for focus information

• Tracking information from Python will be stored locally in database in SQLite

• NodeJS main data transfer point from database to all other system components

• User portal dynamically allocates information from NodeJS into editable table for confirmation
System Architecture

Diagram showing the system architecture with components such as:
- Server
- ReactJS
- User portal
- Node
- Data transfer
- SQLite
- Database
- Backend
- Python
- Graph API
- Communications tracking
- Active Development, Time Tracking
- Clients

The Capstone Experience
Team Vectorform Project Plan Presentation
System Components

• Software Platforms / Technologies
  ▪ SQLite
    o Lightweight SQL relational database platform
  ▪ ReactJS
    o Adaptive user interface creation library
  ▪ NodeJS
    o Asynchronous JavaScript runtime library for scalable networking
  ▪ Python
    o Open-source cross platform programming language
  ▪ Bitbucket
    o Atlassian Git interface for version control
  ▪ Microsoft Teams
    o Communications and Meetings software that we will be tracking
  ▪ Microsoft Visual Studio
    o Professional IDE utilized by Vectorform that we will be tracking
  ▪ Android Studio
    o Professional application development platform used by Vectorform that we will be tracking
Risks

• Connecting Multiple Languages and Applications
  ▪ Description: Our team has minimal experience in linking together multiple languages and applications in a singular system
  ▪ Mitigation: Focusing on getting all the parts connected before developing out more features

• Tracking Active Work Time
  ▪ Description: Need to track time spent in active development, ignore time on projects left open in background
  ▪ Mitigation: Working hard on finding OS level focus tracking and development tool plugins that could help filter for this

• Extended Background Usage
  ▪ Description: Program will need to run for long amounts of time without error in the background, as well as without slowing down hardware
  ▪ Mitigation: Our technologies have been selected to be non-resource intensive and we will be adding endurance testing to our plans
Questions?