Project Weekly Progress Report  
Agile – Scrum

|  |  |
| --- | --- |
| Semester | Enter the Current Semester -3rd 2024Fall |
| Course Code | 1. Enter the course code-CBD-3375(DevOps and Cloud Computing for Canadian Enterprises) |
| Section | Enter your section number -Section 2 |
| Group Name | Enter the group Name -Group F |
| Student names/Student IDs | Enter the team member's name and student number  Sukhjinder Singh-C0908294  Arvind Kumar-C0903042  Rattanjot Singh Tiwana-C0905659  Shaurya Sharma-C0905668 |
| Reporting Week | Week-2 |
| Team Lead for the reporting week | Enter the name of your team lead for this week  Sukhjinder Singh - C0908294 |

Why Weekly Progress Report?   
  
Preparing and submitting weekly progress reports is crucial for stakeholders of a project as it helps them stay informed and involved in the project's development. Progress reports provide stakeholders with a detailed overview of the project's status, including any accomplishments and challenges that were faced during the reporting week. This information allows stakeholders to make informed decisions, provide targeted feedback, and ensure that the project is on track to meet its goals and objectives.

Weekly progress reports also help stakeholders identify any potential issues or risks early on, allowing them to take proactive steps to mitigate them. Ultimately, regular progress reports help build trust and accountability among project stakeholders, leading to more successful outcomes and increased satisfaction for all involved.

Instructions:

Provide detailed information on the progress that you made in the reporting week.

* Minimum Required Length: 3 Pages
* At least one graphical content with figure number and reference to it in the text
* References: Please include APA-style references for each external source you use

# **Progress Made in Reporting** **Week:** (Provide detailed information on the tasks and activities that are completed in the reporting week)

Weekly Report

Weekly Report 2.

Assignments and Events Fulfilled This Week:  
Assigning role and discussing time schedule so as to collaborate and to find which suitable time is good to talk about project after class

As by defined role and responsibilities we are going as follow

Firebase Authentication- Sukhjinder/ Arvind

Locally setting up the environment variables - Ratanjot/ Shaurya

Firebase Project Setup:

* Successfully created a Firebase project and enabled necessary services including Firebase and Authentication.
* Configured Firebase SDK environments.

API Development:

* Set up an Express.js server and initialized it to run on port 3000.
* Created RESTful API endpoints for user registration and data storage:
  + Implemented a /register endpoint to allow new user registrations using Firebase Authentication.
  + Developed a /data endpoint for saving data to Firebase and a corresponding /data/:key endpoint to retrieve stored data.

Firebase Integration:

* Successfully established Firebase database connections and implemented CRUD operations.
* Tested the endpoints using Postman, confirming that data could be saved and retrieved correctly.

Testing and Validation:

* Conducted thorough testing of all API endpoints to ensure they function as expected.
* Validated user authentication processes to ensure secure access to the API.

Code we used

from flask import Flask, render\_template, request, redirect, url\_for, flash, session  
import firebase\_admin  
from firebase\_admin import credentials, auth, storage  
import os  
  
app = Flask(\_\_name\_\_)  
app.secret\_key = os.getenv('FLASK\_SECRET\_KEY', 'default\_key\_for\_dev')  
  
# Initialize Firebase  
cred = credentials.Certificate("C:/Users/gills/Desktop/fileupload1/fileupload1/safelysave-ce984-firebase-adminsdk-sggxw-251af21a63.json")  
firebase\_admin.initialize\_app(cred, {  
 'storageBucket': 'safelysave-ce984.appspot.com'  
})  
  
  
# Home route  
@app.route('/', methods=['GET'])  
def home():  
  
  
 if 'user' in session:  
 return redirect(url\_for('upload\_file'))  
 return render\_template('index.html', page='login')  
  
  
# Login functionality  
@app.route('/login', methods=['POST'])  
def login():  
 email = request.form['email']  
 password = request.form['password']  
  
 try:  
 # Attempt to sign in with the provided credentials  
 user = auth.get\_user\_by\_email(email)  
 # Create a session if the user is found (you may want to handle password verification separately)  
 session['user'] = user.uid  
 return redirect(url\_for('upload\_file'))  
 except Exception as e:  
 flash('Invalid email or password')  
 return redirect(url\_for('home'))  
  
  
# Upload functionality  
@app.route('/upload', methods=['GET', 'POST'])  
def upload\_file():  
 if 'user' not in session:  
 return redirect(url\_for('home'))  
 if request.method == 'POST':  
 if 'file' not in request.files:  
 flash('No file part')  
 return redirect(url\_for('upload\_file'))  
 file = request.files['file']  
 if file.filename == '':  
 flash('No selected file')  
 return redirect(url\_for('upload\_file'))  
 bucket = storage.bucket()  
 blob = bucket.blob(file.filename)  
 blob.upload\_from\_file(file)  
 flash('File uploaded successfully')  
 return redirect(url\_for('upload\_file'))  
 return render\_template('index.html', page='upload')  
  
  
# Logout functionality  
@app.route('/logout', methods=['GET'])  
def logout():  
 session.pop('user', None)  
 return redirect(url\_for('home'))  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug=True)

Suggestions and Future Steps:

1. Enhanced Authentication Features:
   * Multi-Factor Authentication: Implement MFA for added security during user logins.
2. Improved Error Handling and Logging:
   * Centralized Error Handling: Create middleware in Express.js to handle errors consistently and return meaningful messages.
   * Logging Framework: Integrate a logging solution (e.g., Winston or Morgan) to track requests, responses, and errors for easier debugging and monitoring.
3. User Interface Enhancements:
   * If there’s a frontend application, enhance user interface and experience based on user feedback. Ensure seamless interaction with the API.

# **Difficulties Encountered in Reporting Week:**

# Firebase Configuration Challenges:

# Faced initial difficulties with setting up Firebase credentials correctly for the Node.js environment, which led to issues with authentication and data access.

# Required multiple attempts to properly configure Firebase rules, ensuring that they allowed the intended operations without compromising security.

# API Error Handling:

# Encountered issues with proper error handling in the API. Some error messages were vague, making it challenging to diagnose problems.

# Needed to implement more robust logging to track the flow of requests and identify where errors were occurring.

# Testing Complexities:

# Testing the API endpoints with various user scenarios exposed bugs related to data validation and user authentication flows.

# Had to revise the data structure in Firebase to better fit the API requirements, which required additional adjustments in the code.

# Firewall and Network Issues:

# Encountered network connectivity issues when trying to connect to Firebase from the local environment, necessitating adjustments to firewall settings.