COLLEGECODE : 6109

COLLEGENAME : JAYALAKSHMI INSTITUTE OF TECHNOLOGY

DEPARTMENT : COMPUTER SCIENCE AND ENGINEERING

STUDENTNMID :849dbfcce321f459f0148603bb827f00

ROLL NO : 610923104317

DATE : 27-09-2025

Completed the project name as

Phase_3_TECHNOLOGY PROJECT

NAME : IBM-NJ EVENT SCHEDULER APP

SUBMITTED BY,

NAME :MAHENDIRAN M

MOBILE NO :+91 9080640249

Phase3—IBM-NJ EVENT SCHEDULER APP

1. ProjectSetup

- InitializetheprojectrepositoryonGitHub.
- Setup**frontendenvironment**usingReact.js(orHTML, CSS,JSwithTailwindCSS).
- Setup**backendenvironment**withNode.jsandExpress.js.
- Connectbackendwithadatabase(MongoDB/MySQL).
- Installdependencies(ReactRouter,Axios,Express,JWT,bcrypt,Mongoose/Sequelize).
- Initializetheproject environment usinga modernframework(e.g.,ReactNative/Flutterformobile,orReact/ Vue for web).
- Setupdevelopmenttools,dependencies,and folderstructure.
- Configureenvironmentfilesandpackagemanagers.
- Integrateataskmanager(likeTrello,Jira)forsprinttracking.

2. CoreFeaturesImplementation

- **UserAuthentication**:Login,Register,andLogout withsecurepasswordhashing.
- **Event Management**:
  - Createanevent(title,description,date,time, venue).
  - Updateorcancelanexistingevent.
  - Vieweventdetailsinadashboardorcalendarview.
- **EventReminders**:Notificationsystemtoalertusersbeforeeventstart.
- **ParticipantManagement**:Optiontoinviteparticipants(email/notification).
- **UserRegistration/Login**(Authenticationsystem)
- **CreateEvents**:Userscanscheduleeventsbyinputtingtitle, description, time, date,andlocation.
- **Edit/DeleteEvents**
- **ViewScheduledEvents**inalistorcalendarview.
- **Reminder/NotificationSystem**(ifpossiblewithinMVPscope)
- BasicUI/UX fornavigationbetweenviews/screens.

3. DataStorage(LocalState/Database)

- **FrontendLocalState**:
  - Storetemporarydata likeevent forminputs,authenticationtokens,andUIstatesusingReact's useState/useContext or Redux.
- **Database**:
  - Usertable/collection(ID,username,email,password).
  - Eventstable/collection(ID,title, description,date,time,venue,participants).
  - Notifications/reminderstable/collection(linkedtoevents).

- Use**localstatemanagement** (e.g.,Redux,React ContextAPI,orProviderforFlutter)formanagingtemporary user and event data.

- Setup**persistentstorage**:

- **Frontend-onlyMVP**:Use`localStorage`or`AsyncStorage`forstoringdatatemporarily.
- **Full-stackMVP**:Integratewithabackend(e.g.,Firebase,Node.js+MongoDB)foruserandevent data persistence.

- Defineandstructuredatamodels(Event,User).

4. TestingCoreFeatures

- Test**UserAuthentication**→userscanregister/loginsuccessfully.
- Test**EventCRUDoperations**→create,update,deleteevents.
- Test **EventReminders**→notificationstriggeratcorrecttimes.
- Test**Multi-userfunctionality** →eventssynccorrectlybetweenusers.
- UnittestingwithJest/MochaforbackendAPIs.
- Writeunit testsforcorefunctionalitieslikeeventcreation,deletion,and validation.
- Performmanualtesting ofuserflows:
- Register→Login→ScheduleEvent→ViewEvent→Edit/Delete

- Usetestingtools(e.g., Jest, FlutterTest,PostmanforAPItesting).
- Bugtrackinganditerationbasedontestresults.

5. VersionControl(GitHub)

- Maintainrepositorywith**branches**foreachfeature(e.g.,`auth-feature`, `event-feature`).
- Regularcommitswithmeaningfulmessages.
- Use**GitHubProjects/Issues**fortasktracking.
- Enable**GitHubActions**forautomatedtestinganddeploymentpipeline.
- InitializeaGitrepositoryand pushcodeto GitHub.
- Usebranchingstrategy(e.g., `main`,`dev`,`feature/*`)forcollaborativedevelopment.
- Regularcommitswithmeaningfulmessages.
- Ensureacleanandwell-documentedcommithistory.