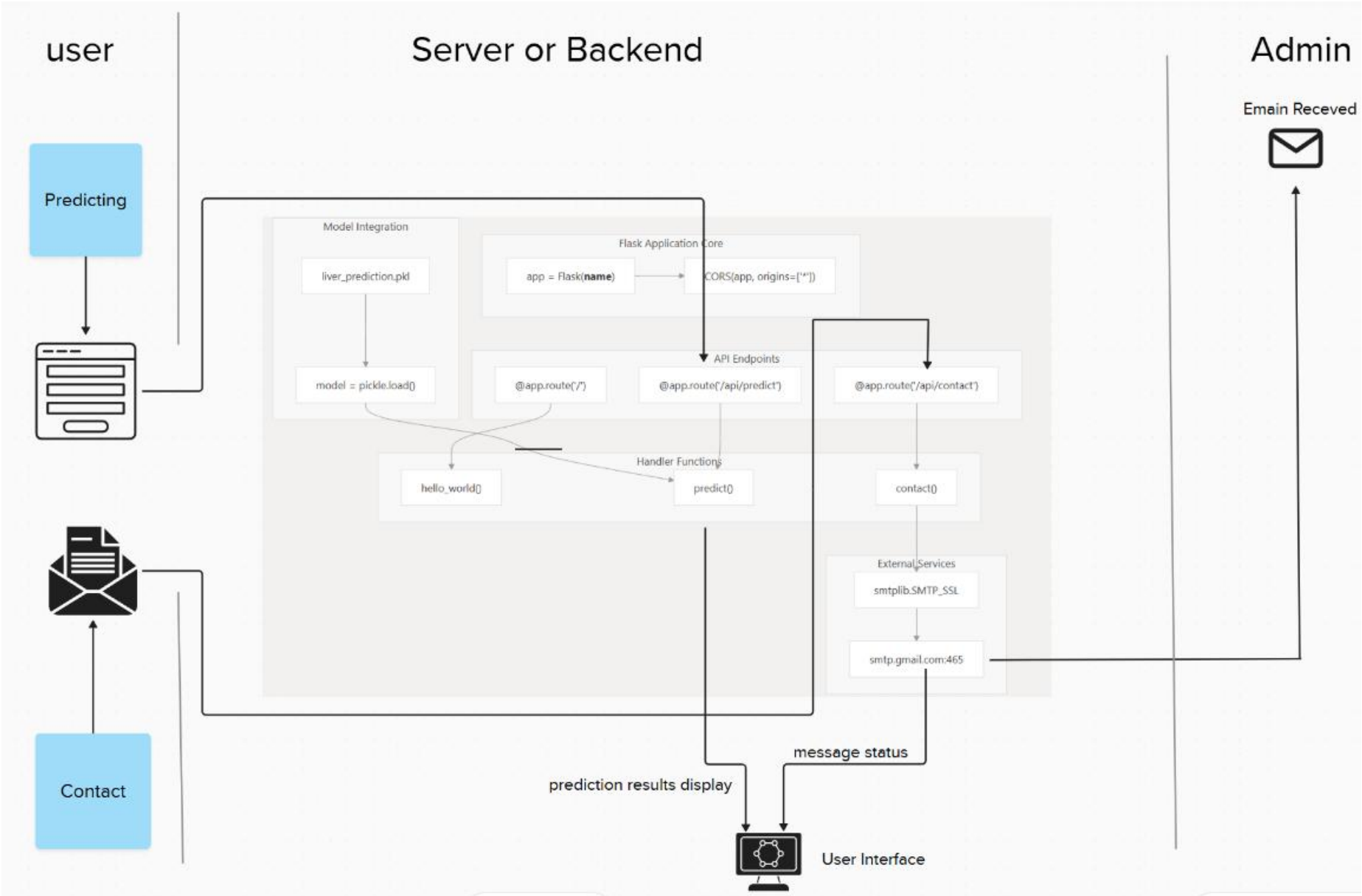
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	16 June 2025
Team ID	LTVIP2025TMID35624
Project Name	Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques
Maximum Marks	4 Marks

Technical Architecture:


user
Server or Backend
Admin
Emain Receved
Predicting
Model Integration
liver_prediction.pkl
model = pickle.load()
Flask Application Core
app = Flask(name)
CORS(app, origins=["*"])
API Endpoints
@app.route('/')
@app.route('/api/predict')
@app.route('/api/contact')
Handler Functions
hello_world()
predict()
contact()
External Services
smtplib.SMTP_SSL
smtp.gmail.com:465
Contact
message status
prediction results display
User Interface

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web-based UI for patients and doctors to enter data and view results and contact with admin	HTML, CSS, JavaScript
2.	Application Logic-1	Backend logic to manage routing, form handling, API endpoints	Python (Flask)
3.	Application Logic-2	Contact form handler that sends messages to admin via Gmail SMTP	Python smtplib, email.message
4.	Application Logic-3	ML model prediction handler: Preprocess data, predict, return response	Flask, NumPy, Pandas
5.	External API-1	Not applicable in this version	-
6.	External API-2	Gmail API used to send user messages to admin	Google Gmail SMTP
7.	Machine Learning Model	Predict liver cirrhosis using health data	Random Forest Classifier (scikit-learn)
8.	Infrastructure (Server / Cloud)	App deployed on local server and cloud (Docker-ready)	Localhost / Flask / Docker-ready

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Used frameworks and libraries are open-source	Flask, scikit-learn, Pandas, NumPy, Matplotlib & Seaborn
2.	Security Implementations	Input validation, secure SMTP mail handling, CORS headers in backend	Gmail Auth, Flask-CORS.
3.	Scalable Architecture	Backend is modular, ML model is containerizable and API is RESTful	3-tier architecture (Frontend, API, Model)
4.	Availability	Flask API deployable with Docker support ensures availability in onrender.com server Html/Css deployable in netlify.app	Docker
5.	Performance	Optimized preprocessing and fast inference (<2 sec); prediction via local model	Flask API, pickle load, low latency model

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d